Blood analysis - pathprint v0.3 beta4 - hpc111 version

Gabriel Altschuler

October 3, 2011

In this document we will use the blood normal and leukemic lineages to demonstrate the use of the pathway fingerprint to construct phylogneies.
The required data and metadata is contained within the R package `pathprint`. We will use the `pathprint.v.0.3.beta3` build in this session. The next step is to replicate the blood analysis for the new pathprint package and integrate the zebrafish data wherever possible.

1 Human blood lineage

An initial analysis will be on the human hematopoiesis data published in *Novershtern et al. Densely Interconnected Transcriptional Circuits Control Cell States in Human Hematopoiesis. Cell (2011).* This is contained within the GEO record GSE24759. First we need to source the pathprint package and load the data libraries.
The metadata can be extracted from the pathprint metadata matrix

```
> library(pathprint.v0.3.beta4)
> data(GEO.metadata.matrix)
> GSE24759.meta<-GEO.metadata.matrix[
    GEO.metadata.matrix$GSE %in% "GSE24759",]
> GSE24759.meta$cellType<-sapply(GSE24759.meta$Characteristics,
           function(x){unlist(strsplit(x, split = ";"))[[1]]})
> GSE24759.meta$cellType<-gsub("cell type: ", "", GSE24759.meta$cellType)
> GSE24759.cellTypes<-levels(as.factor(GSE24759.meta$cellType))
```

The fingerprints can be extracted from the fingerprint matrix and a consensus fingerprint constructed for each of the cell types.

```
> threshold <- 0.5
> data(GEO.fingerprint.matrix)
> GSE24759.data<-GEO.fingerprint.matrix[,GSE24759.meta$GSM]
> GSE24759.consensus<-sapply(GSE24759.cellTypes, function(x){
    consensusFingerprint(GEO.fingerprint.matrix[,
        GSE24759.meta$GSM[GSE24759.meta$cellType == x]],
```

```
     threshold = threshold)
     })
> GSE24759.consensus[1:5, 1:2]
                                                Basophils
Glycolysis / Gluconeogenesis (KEGG)                    -1
Citrate cycle (TCA cycle) (KEGG)                       -1
Pentose phosphate pathway (KEGG)                       -1
Pentose and glucuronate interconversions (KEGG)         1
Fructose and mannose metabolism (KEGG)                 -1
                                                CD4+ Central Memory
Glycolysis / Gluconeogenesis (KEGG)                              -1
Citrate cycle (TCA cycle) (KEGG)                                 -1
Pentose phosphate pathway (KEGG)                                 -1
Pentose and glucuronate interconversions (KEGG)                   0
Fructose and mannose metabolism (KEGG)                           -1
```

Next the fingerprint matrix will be used to construct a optimum parsimony phylogentic tree. This requires the packages `ape` and `phangorn`. One caveat is that the server installed version of `ape` is not compatible with `phangorn` so if running on the server a locally installed updated version is required.
A bootstrapped tree will be constructed. The consensus and the bootstrap values will be plotted

```
> try(library(ape, lib.loc = .libPaths()[3]))
> library(ape)
> library(phangorn)
> try(library(multicore))
> #try(library(doMC))
> #try(registerDoMC(cores = 10))
> # define cost matrix for transitions between two states
> CM<-matrix(c(0,1,2,1,0,1,2,1,0), ncol = 3)
> dimnames(CM) <- list(c(-1,0,1), c(-1,0,1))
> GSE24759.dat <- phyDat(t(GSE24759.consensus), type = "USER", levels = c(-1,0,1))
> GSE24759.dist <- dist.hamming(GSE24759.dat)
> # construct trees
> GSE24759.NJ.tree <- NJ(GSE24759.dist)
> #GSE24759.parsimony <- pratchet(
> #  GSE24759.dat, start = GSE24759.NJ.tree, k = 50,
> #  method = "sankoff", cost = CM, trace = 0, np = 1,all = TRUE)
>
> # Bootstrap
> GSE24759.parsimony.boot <- bootstrap.phyDat(
   GSE24759.dat, bs = 100, pratchet, start = GSE24759.NJ.tree, k = 100,
   method = "sankoff", cost = CM, trace = 0, np = 1, all = TRUE)
> # Combine multiple bootstrap trees
> GSE24759.parsimony.boot<-c(
```

```
   GSE24759.parsimony.boot[
     lapply(GSE24759.parsimony.boot, class) == "phylo"
     ],
   unlist(GSE24759.parsimony.boot[
     lapply(GSE24759.parsimony.boot, class) == "multiPhylo"
     ],
          recursive = FALSE)
   )
> # Convert to cladewise ordering of edges
> GSE24759.parsimony.boot<-lapply(GSE24759.parsimony.boot, reorder, "cladewise")
> class(GSE24759.parsimony.boot)<-"multiPhylo"
> # Create consensus tree
> GSE24759.parsimony.consensus<-consensus(GSE24759.parsimony.boot, p = 0.5)
> # Calculate bootstrap scores
> GSE24759.parsimony.consensus$node.label<-round((100*prop.clades(
   GSE24759.parsimony.consensus, GSE24759.parsimony.boot)
                       )/length(GSE24759.parsimony.boot))
> for (i in 1:length(GSE24759.parsimony.boot)){
   GSE24759.parsimony.boot[[i]]$node.label<-(100*prop.clades(
     GSE24759.parsimony.boot[[i]], GSE24759.parsimony.boot)
                       )/length(GSE24759.parsimony.boot)
 }
```

We can now either select the tree with highest summed bootstrap scores or the best parsimony scores. Here we will use the best parsimony score. These trees will be rooted

```
> GSE24759.bootstrap.scores<-data.frame(
   pScore = sapply(GSE24759.parsimony.boot, attr, "pscore"),
   sumBoot = sapply(GSE24759.parsimony.boot, function(x){
     sum(x$node.label)
     })
   )
> # Show top ordered trees
> head(GSE24759.bootstrap.scores[
   order(GSE24759.bootstrap.scores$pScore, -GSE24759.bootstrap.scores$sumBoot),
   ])
    pScore  sumBoot
199   1575 2073.859
202   1575 2059.336
200   1575 2053.527
204   1575 2045.228
201   1575 2041.494
206   1575 2030.705
> head(GSE24759.bootstrap.scores[
   order(GSE24759.bootstrap.scores$sumBoot, decreasing = TRUE),
   ])
```

```
    pScore  sumBoot
172   1650 2105.809
14    1792 2085.062
199   1575 2073.859
170   1650 2073.029
202   1575 2059.336
171   1650 2056.017
> GSE24759.parsimony.top<-GSE24759.parsimony.boot[[
   order(GSE24759.bootstrap.scores$pScore, -GSE24759.bootstrap.scores$sumBoot)[1]
   ]]
> GSE24759.parsimony.top$node.label<-round(GSE24759.parsimony.top$node.label)
> # Root trees to HSC
>
> try(GSE24759.parsimony.top<-root(
     GSE24759.parsimony.top, 21, resolve.root = TRUE))


> plot(GSE24759.parsimony.top, show.node.label = FALSE, label.offset = 1)
> nodelabels(GSE24759.parsimony.top$node.label, bg = "white", cex = 0.75)


> plot(GSE24759.parsimony.consensus, show.node.label = FALSE, label.offset = 1)
> nodelabels(GSE24759.parsimony.consensus$node.label, bg = "white", cex = 0.75)
```

2 Mouse hematopoiesis

The GEO dataset GSE6506 profiles hematopoietic stem and progenitor cells during mouse blood development. It would be interesting to know how these cell types relate in terms of their pathway fingerprints.

```
> GSE6506.meta<-GEO.metadata.matrix[
     GEO.metadata.matrix$GSE %in% "GSE6506",]
> GSE6506.meta$cellType<-sapply(GSE6506.meta$Source,
            function(x){unlist(strsplit(x, split = " isolated"))[[1]]})
> GSE6506.meta$cellType<-gsub(" activated with an LPS treatment and", "", GSE6506.meta$c
> GSE6506.cellTypes<-levels(as.factor(GSE6506.meta$cellType))
```

The fingerprints can be extracted from the fingerprint matrix and a consensus fingerprint constructed for each of the cell types.

```
> GSE6506.data<-GEO.fingerprint.matrix[,GSE6506.meta$GSM]
> GSE6506.consensus<-sapply(GSE6506.cellTypes, function(x){
     consensusFingerprint(GEO.fingerprint.matrix[,
         GSE6506.meta$GSM[GSE6506.meta$cellType == x]],
     threshold = threshold)
     })
> GSE6506.consensus[1:5, 1:5]
                                              B-Cells
Glycolysis / Gluconeogenesis (KEGG)                -1
Citrate cycle (TCA cycle) (KEGG)                   -1
Pentose phosphate pathway (KEGG)                   -1
Pentose and glucuronate interconversions (KEGG)     0
Fructose and mannose metabolism (KEGG)             -1
                                              CD4+ T-cell
Glycolysis / Gluconeogenesis (KEGG)                 0
Citrate cycle (TCA cycle) (KEGG)                    0
Pentose phosphate pathway (KEGG)                    0
Pentose and glucuronate interconversions (KEGG)    -1
Fructose and mannose metabolism (KEGG)              0
                                              CD4+ T-cell naive
Glycolysis / Gluconeogenesis (KEGG)                        -1
Citrate cycle (TCA cycle) (KEGG)                           -1
Pentose phosphate pathway (KEGG)                           -1
Pentose and glucuronate interconversions (KEGG)            -1
Fructose and mannose metabolism (KEGG)                     -1
                                              CD8+ T-cell
Glycolysis / Gluconeogenesis (KEGG)                -1
Citrate cycle (TCA cycle) (KEGG)                    0
Pentose phosphate pathway (KEGG)                    0
Pentose and glucuronate interconversions (KEGG)    -1
Fructose and mannose metabolism (KEGG)              0
```

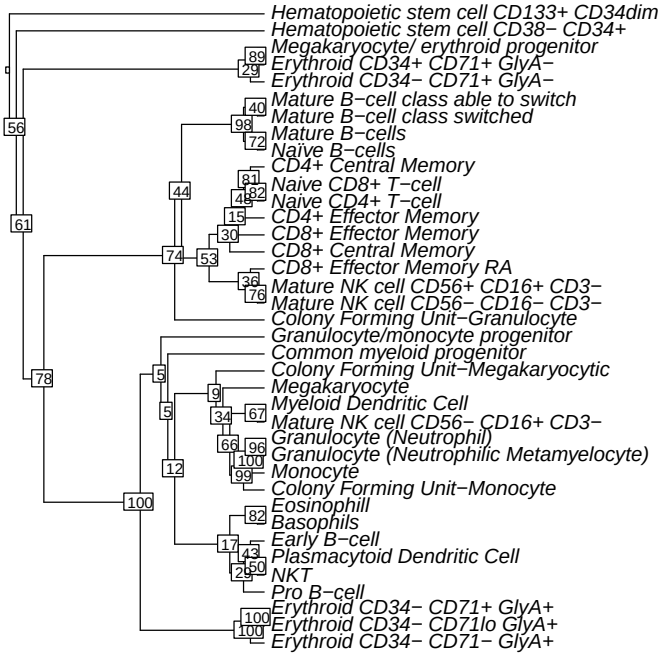

Figure 1: Maximum parsimony tree found for GSE24759 cell types, numbers indicate bootstrap values

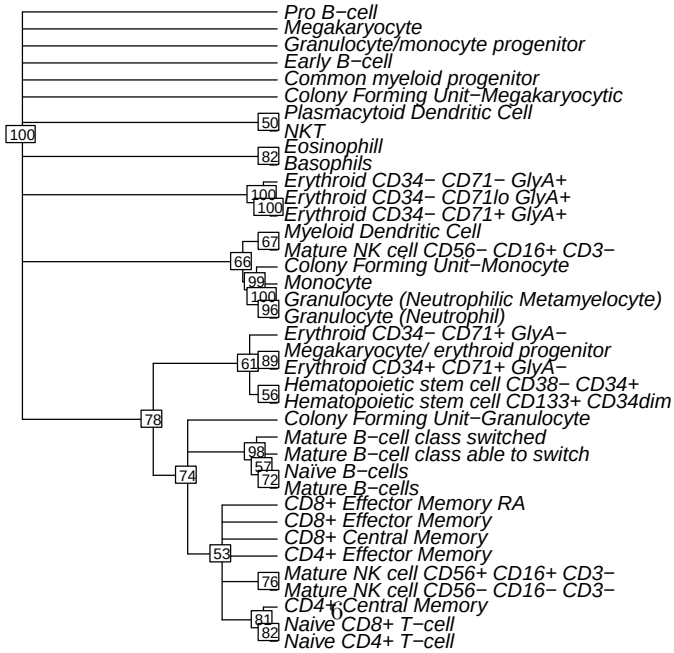

Figure 2: Consensus parsimony tree for GSE24759 cell types from bootstraps, numbers indicate bootstrap values

```
                                                CD8+ T-cell naive
Glycolysis / Gluconeogenesis (KEGG)                            -1
Citrate cycle (TCA cycle) (KEGG)                               -1
Pentose phosphate pathway (KEGG)                               -1
Pentose and glucuronate interconversions (KEGG)                -1
Fructose and mannose metabolism (KEGG)                         -1
> GSE6506.inform<-rownames(GSE6506.consensus)[apply(GSE6506.consensus, 1, sd) > 0]


> GSE6506.dat <- phyDat(t(GSE6506.consensus), type = "USER", levels = c(-1,0,1))
> GSE6506.dist <- dist.hamming(GSE6506.dat)
> # construct trees
> GSE6506.NJ.tree <- NJ(GSE6506.dist)
> GSE6506.parsimony.boot <- bootstrap.phyDat(
   GSE6506.dat, bs = 100, pratchet, start = GSE6506.NJ.tree, k = 50,
   method = "sankoff", cost = CM, trace = 0, np = 1, all = TRUE)
> GSE6506.parsimony.boot<-c(
   GSE6506.parsimony.boot[
     lapply(GSE6506.parsimony.boot, class) == "phylo"
     ],
   unlist(GSE6506.parsimony.boot[
     lapply(GSE6506.parsimony.boot, class) == "multiPhylo"
     ],
          recursive = FALSE)
   )
> # Convert to cladewise ordering of edges
> GSE6506.parsimony.boot<-lapply(GSE6506.parsimony.boot, reorder, "cladewise")
> class(GSE6506.parsimony.boot)<-"multiPhylo"
> # Create consensus tree
> GSE6506.parsimony.consensus<-consensus(GSE6506.parsimony.boot, p = 0.5)
> # Add bootstrap scores
> GSE6506.parsimony.consensus$node.label<-round((100*prop.clades(
   GSE6506.parsimony.consensus, GSE6506.parsimony.boot)
                       )/length(GSE6506.parsimony.boot))
> for (i in 1:length(GSE6506.parsimony.boot)){
   GSE6506.parsimony.boot[[i]]$node.label<-(100*prop.clades(
     GSE6506.parsimony.boot[[i]], GSE6506.parsimony.boot)
                       )/length(GSE6506.parsimony.boot)
 }
```

As before, we can now either select the tree with highest summed bootstrap scores or the best parsimony scores. Here we will use the best parsimony score.

```
> GSE6506.bootstrap.scores<-data.frame(
   pScore = sapply(GSE6506.parsimony.boot, attr, "pscore"),
   sumBoot = sapply(GSE6506.parsimony.boot, function(x){
```

```
     sum(x$node.label)
     })
   )
> # Show top ordered trees
> head(GSE6506.bootstrap.scores[
   order(GSE6506.bootstrap.scores$pScore, -GSE6506.bootstrap.scores$sumBoot),
   ])
   pScore  sumBoot
28    723 400.7812
24    731 430.4688
20    746 391.4062
71    749 408.5938
61    750 410.1562
56    753 400.0000
> head(GSE6506.bootstrap.scores[
   order(GSE6506.bootstrap.scores$sumBoot, decreasing = TRUE),
   ])
   pScore  sumBoot
81    836 433.5938
59    780 432.0312
24    731 430.4688
41    830 430.4688
43    810 430.4688
77    827 430.4688
> # Select tree with best parsimony score
> GSE6506.parsimony.top<-GSE6506.parsimony.boot[[
   order(GSE6506.bootstrap.scores$pScore, -GSE6506.bootstrap.scores$sumBoot)[1]
   ]]
> GSE6506.parsimony.top$node.label<-round(GSE6506.parsimony.top$node.label)
> # Root trees to HSC
> try(GSE6506.parsimony.top<-root(
     GSE6506.parsimony.top, 7, resolve.root = TRUE))
> try(GSE6506.parsimony.consensus<-root(
     GSE6506.parsimony.consensus, 7, resolve.root = TRUE))


> plot(GSE6506.parsimony.top, show.node.label = FALSE, label.offset = 0)
> nodelabels(GSE6506.parsimony.top$node.label, bg = "white", cex = 0.75)


> plot(GSE6506.parsimony.consensus, show.node.label = FALSE, label.offset = 0)
> nodelabels(GSE6506.parsimony.consensus$node.label, bg = "white", cex = 0.75)
```

3 Hematopoietic Stem Cells during Zebrafish Development

The GEO dataset GSE7658 profiles hematopoietic stem and progenitor cells during regular development in the zebrafish embryo. gata1-GFP+/+(18 somites), lmo2-GFP+/+ (18 somites and 35 hpf)1 and cd41-GFP+/+ (35 hpf) cells from transgenic embryos were individually separated from GFP-/- cells by flow cytometry at the indicated stages. It would be interesting to know how these cell types relate to the human blood tree.

```
> GSE7658.meta<-GEO.metadata.matrix[
     GEO.metadata.matrix$GSE %in% "GSE7658",]
> GSE7658.meta$cellType<-sapply(GSE7658.meta$Title,
             function(x){unlist(strsplit(x, split = "_"))[[1]]})
> GSE7658.meta$cellType<-gsub("zebrafish-35hpf-cd41-sorted-", "", GSE7658.meta$cellType)
> GSE7658.cellTypes<-levels(as.factor(GSE7658.meta$cellType))
> GSE7658.meta$cellType_main<-substr(GSE7658.meta$cellType, 1, 8)
> GSE7658.cellType_main<-levels(as.factor(GSE7658.meta$cellType_main))
```

The fingerprints can be extracted from the fingerprint matrix and a consensus fingerprint constructed for each of the cell types.

```
> GSE7658.data<-GEO.fingerprint.matrix[,GSE7658.meta$GSM]
> colnames(GSE7658.data)<-GSE7658.meta$cellType
> head(GSE7658.data)
                                              GFP-pos+3
Glycolysis / Gluconeogenesis (KEGG)                   0
Citrate cycle (TCA cycle) (KEGG)                     -1
Pentose phosphate pathway (KEGG)                      1
Pentose and glucuronate interconversions (KEGG)       1
Fructose and mannose metabolism (KEGG)                0
Galactose metabolism (KEGG)                           1
                                              GFP-neg-3
Glycolysis / Gluconeogenesis (KEGG)                   0
Citrate cycle (TCA cycle) (KEGG)                     -1
Pentose phosphate pathway (KEGG)                      0
Pentose and glucuronate interconversions (KEGG)      -1
Fructose and mannose metabolism (KEGG)                0
Galactose metabolism (KEGG)                          -1
                                              GFP-pos+6
Glycolysis / Gluconeogenesis (KEGG)                   1
Citrate cycle (TCA cycle) (KEGG)                      1
Pentose phosphate pathway (KEGG)                      1
Pentose and glucuronate interconversions (KEGG)       1
Fructose and mannose metabolism (KEGG)                0
Galactose metabolism (KEGG)                           1
```

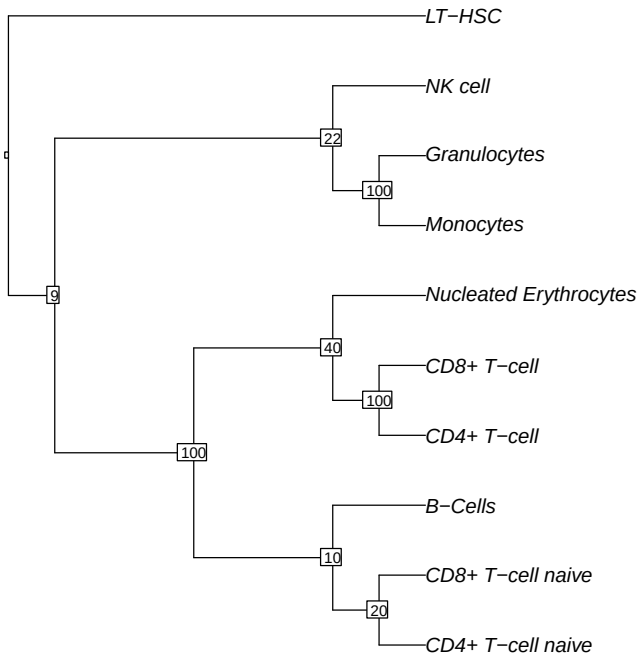

Figure 3: Maximum parsimony tree with highest bootstrap values for GSE6506 cell types, numbers indicate bootstrap values

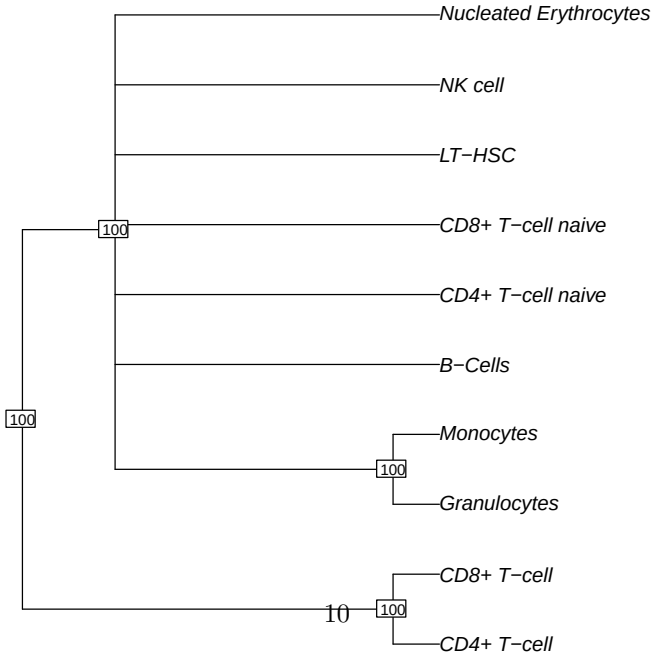

Figure 4: Consensus parsimony tree for GSE6506 cell types from bootstraps, numbers indicate bootstrap values

```
                                          GFP-neg-6
Glycolysis / Gluconeogenesis (KEGG)              -1
Citrate cycle (TCA cycle) (KEGG)                 -1
Pentose phosphate pathway (KEGG)                  0
Pentose and glucuronate interconversions (KEGG)  -1
Fructose and mannose metabolism (KEGG)            0
Galactose metabolism (KEGG)                      -1
                                          GFP-pos+8
Glycolysis / Gluconeogenesis (KEGG)               0
Citrate cycle (TCA cycle) (KEGG)                  0
Pentose phosphate pathway (KEGG)                  0
Pentose and glucuronate interconversions (KEGG)   0
Fructose and mannose metabolism (KEGG)            0
Galactose metabolism (KEGG)                       1
                                          GFP-neg-8
Glycolysis / Gluconeogenesis (KEGG)              -1
Citrate cycle (TCA cycle) (KEGG)                 -1
Pentose phosphate pathway (KEGG)                  0
Pentose and glucuronate interconversions (KEGG)  -1
Fructose and mannose metabolism (KEGG)            0
Galactose metabolism (KEGG)                      -1
                                          GFP-pos+9
Glycolysis / Gluconeogenesis (KEGG)               1
Citrate cycle (TCA cycle) (KEGG)                  1
Pentose phosphate pathway (KEGG)                  0
Pentose and glucuronate interconversions (KEGG)   1
Fructose and mannose metabolism (KEGG)            0
Galactose metabolism (KEGG)                       1
                                          GFP-neg-9
Glycolysis / Gluconeogenesis (KEGG)              -1
Citrate cycle (TCA cycle) (KEGG)                 -1
Pentose phosphate pathway (KEGG)                  0
Pentose and glucuronate interconversions (KEGG)  -1
Fructose and mannose metabolism (KEGG)            0
Galactose metabolism (KEGG)                      -1
```

```
> GSE7658.dat <- phyDat(t(GSE7658.data), type = "USER", levels = c(-1,0,1))
> GSE7658.dist <- dist.hamming(GSE7658.dat)
> # construct trees
> GSE7658.NJ.tree <- NJ(GSE7658.dist)
> GSE7658.parsimony.boot <- bootstrap.phyDat(
   GSE7658.dat, bs = 100, pratchet, start = GSE7658.NJ.tree, k = 50,
   method = "sankoff", cost = CM, trace = 0, np = 1, all = TRUE)
> GSE7658.parsimony.boot<-c(
   GSE7658.parsimony.boot[
```

```
    lapply(GSE7658.parsimony.boot, class) == "phylo"
    ],
  unlist(GSE7658.parsimony.boot[
    lapply(GSE7658.parsimony.boot, class) == "multiPhylo"
    ],
         recursive = FALSE)
  )
> # Convert to cladewise ordering of edges
> GSE7658.parsimony.boot<-lapply(GSE7658.parsimony.boot, reorder, "cladewise")
> class(GSE7658.parsimony.boot)<-"multiPhylo"
> # Create consensus tree
> GSE7658.parsimony.consensus<-consensus(GSE7658.parsimony.boot, p = 0.5)
> # Add bootstrap scores
> GSE7658.parsimony.consensus$node.label<-round((100*prop.clades(
  GSE7658.parsimony.consensus, GSE7658.parsimony.boot)
                    )/length(GSE7658.parsimony.boot))
> for (i in 1:length(GSE7658.parsimony.boot)){
  GSE7658.parsimony.boot[[i]]$node.label<-(100*prop.clades(
    GSE7658.parsimony.boot[[i]], GSE7658.parsimony.boot)
                    )/length(GSE7658.parsimony.boot)
 }
```

As before, we can now either select the tree with highest summed bootstrap scores or the best parsimony scores. Here we will use the best parsimony score.

```
> GSE7658.bootstrap.scores<-data.frame(
  pScore = sapply(GSE7658.parsimony.boot, attr, "pscore"),
  sumBoot = sapply(GSE7658.parsimony.boot, function(x){
    sum(x$node.label)
    })
  )
> # Show top ordered trees
> head(GSE7658.bootstrap.scores[
  order(GSE7658.bootstrap.scores$pScore, -GSE7658.bootstrap.scores$sumBoot),
  ])
  pScore  sumBoot
4    1109 541.7476
93   1110 541.7476
50   1127 541.7476
3    1131 538.8350
41   1133 541.7476
47   1140 538.8350
> head(GSE7658.bootstrap.scores[
  order(GSE7658.bootstrap.scores$sumBoot, decreasing = TRUE),
  ])
```

```
   pScore  sumBoot
1    1235 541.7476
4    1109 541.7476
5    1141 541.7476
6    1148 541.7476
9    1203 541.7476
11   1216 541.7476
> # Select tree with best parsimony score
> GSE7658.parsimony.top<-GSE7658.parsimony.boot[[
   order(GSE7658.bootstrap.scores$pScore, -GSE7658.bootstrap.scores$sumBoot)[1]
   ]]
> GSE7658.parsimony.top$node.label<-round(GSE7658.parsimony.top$node.label)


> plot(GSE7658.parsimony.top, show.node.label = FALSE, label.offset = 0)
> nodelabels(GSE7658.parsimony.top$node.label, bg = "white", cex = 0.75)


> plot(GSE7658.parsimony.consensus, show.node.label = FALSE, label.offset = 0)
> nodelabels(GSE7658.parsimony.consensus$node.label, bg = "white", cex = 0.75)
```

4 Combining trees

Can we combine the blood lineage data from the different species? Rather than trying to co-cluster the samples into a single tree, we will base use the human tree as a backbone and find the optimum (maximum parsimomy) position to insert each of the zebrafish samples, one-by-one. This will be achieved by inserting branches corresponding to each of the zebrafish samples at all possible positions on the human tree and finding the orientation with the maximum parsimony. Rather than using all pathways we will use only the pathways that are informative to the human tree.

```
> optimP.branch<-function(testMatrix, treeMatrix, tree){
   if(class(tree)!="phylo") stop("tree should be an object of class 'phylo.'")
   if(!exists("CM")) stop("no cost matrix loaded")
   if(!all.equal(rownames(testMatrix), rownames(treeMatrix))) stop("matrices not matched
   tip.color<-c(rep("black", ncol(treeMatrix)), "red")
   top.tree<-list()
   min<-list()
   for (i in 1:ncol(testMatrix)){
     name<-colnames(testMatrix)[i]
     dat.add<-phyDat(
       t(cbind(treeMatrix, testMatrix[,i,drop = FALSE])),
       type = "USER", levels = c(-1,0,1)
       )
     tree.add<-add.everywhere(tree, name)
     p<-parsimony(tree.add, dat.add, method = "sankoff", cost = CM)
     min<-which(p == min(p))
     for (j in 1:length(min)){
       names<-names(top.tree)
       n<-min[j]
       top.tree.add<-tree.add[[n]]
       top.tree.add$tip.color<-tip.color
       top.tree<-append(top.tree, list(top.tree.add))
       names(top.tree)<-c(names, paste(name, j, sep = "_"))
       }
     }
   class(top.tree)<-"multiPhylo"
   return(top.tree)
   }
> add.everywhere<-function(tree,tip.name){
 # This is a function based on a script written by Liam Revell 2011
   if(!require(ape)) stop("function needs 'ape' package.")
   if(class(tree)!="phylo") stop("tree should be an object of class 'phylo.'")
         # Convert to cladewise as pruningwise trees do not work
   tree<-reorder(tree, "cladewise")
   tree<-unroot(tree) # unroot tree
```

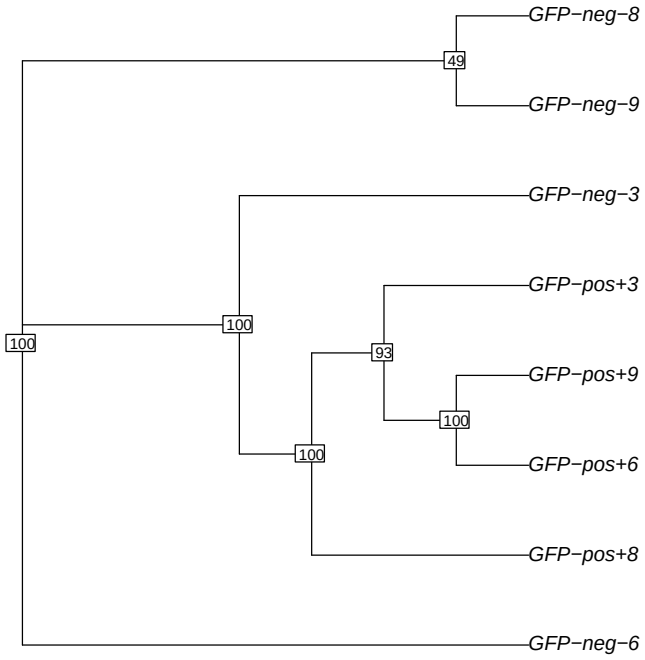

Figure 5: Maximum parsimony tree with highest bootstrap values for GSE7658 cell types, numbers indicate bootstrap values

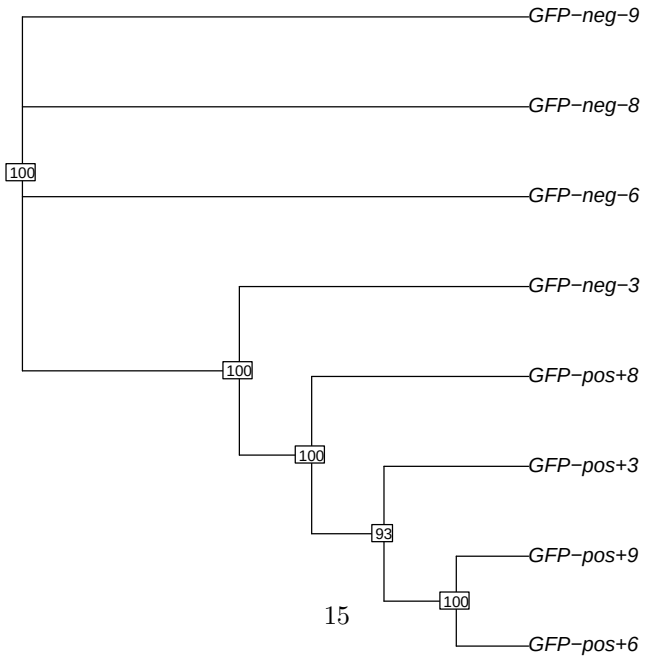

Figure 6: Consensus parsimony tree for GSE7658 cell types from bootstraps, numbers indicate bootstrap values

```
        tree$edge.length<-rep(1,nrow(tree$edge)) # set all edge lengths to 1.0
  # create new tip
        new.tip<-list(edge=matrix(c(2L,1L),1,2),tip.label=tip.name,edge.length=1,Nnode=
        class(new.tip)<-"phylo"
  # add the new tip to all edges of the tree
        trees<-list(); class(trees)<-"multiPhylo"
        for(i in 1:nrow(tree$edge)){
    try(trees[[i]]<-bind.tree(tree,new.tip,where=tree$edge[i,2],position=0.5))
              try(trees[[i]]$edge.length<-NULL)
        }
        trees<-trees[sapply(trees, class) == "phylo"]
  return(trees)
}
```

```
> GSE24759.inform<-rownames(GSE24759.consensus)[apply(GSE24759.consensus, 1, sd) > 0]
> GSE7658.inform<-rownames(GSE7658.data)[apply(GSE7658.data, 1, sd) > 0]
> GSE24759.GSE7658.inform<-intersect(GSE24759.inform, GSE7658.inform)
```

```
> GSE7658.branches.inform<-optimP.branch(GSE7658.data[GSE24759.GSE7658.inform,],
                                         GSE24759.consensus[GSE24759.GSE7658.inform,],
                                         GSE24759.parsimony.top
                                         )
```

```
> l<-length(GSE7658.branches.inform)
> par(mfcol = c(((l %/% 3)+as.numeric((l %% 3 > 0))), ((l %/% 3)+as.numeric((l %% 3 > 0)
> for (i in 1:length(GSE7658.branches.inform)){
   plot(GSE7658.branches.inform[[i]],
        tip.color = GSE7658.branches.inform[[i]]$tip.color)
}
```

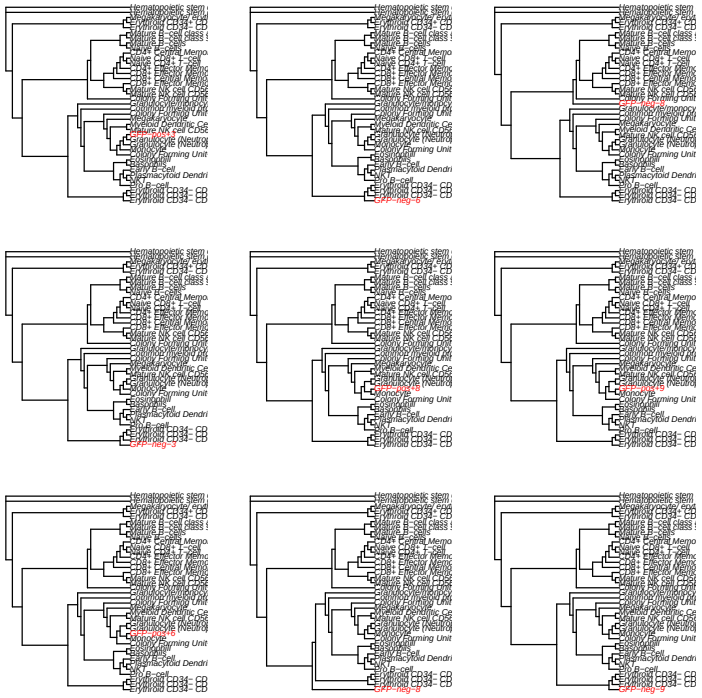

Figure 7: Maximum parsimony trees for each GSE7658 cell type integrated into the GSE24759 tree

5 Combining based on major cell types

An Alternative is to combine the data using only the major cell types

```
> GSE24759.meta$type<-NA
> GSE24759.meta$type[
   grep("CD4+ Effector Memory", GSE24759.meta$cellType, fixed = TRUE)
                  ]<-"CD4+_activated"
> GSE24759.meta$type[
   setdiff(grep("CD8+ Effector Memory", GSE24759.meta$cellType, fixed = TRUE),
           grep("CD8+ Effector Memory RA", GSE24759.meta$cellType, fixed = TRUE))
                  ]<-"CD8+_activated"
> GSE24759.meta$type[
   grep("Naive CD4+ T-cell", GSE24759.meta$cellType, fixed = TRUE)
                  ]<-"CD4+_naive"
> GSE24759.meta$type[
   grep("Naive CD8+ T-cell", GSE24759.meta$cellType, fixed = TRUE)
                   ]<-"CD8+_naive"
```

```
> GSE24759.meta$type[
   grep("Mature B-cells", GSE24759.meta$cellType, fixed = TRUE)
                    ]<-"B-Cell"
> GSE24759.meta$type[
   grep("Mature NK cell", GSE24759.meta$cellType, fixed = TRUE)
                    ]<-"NK"
> GSE24759.meta$type[
   grep("Hematopoietic stem cell_CD133+", GSE24759.meta$cellType, fixed = TRUE)
                   ]<-"LT_HSC"
> GSE24759.meta$type[
   setdiff(grep("Monocyte", GSE24759.meta$cellType, fixed = TRUE),
           grep("Colony Forming Unit-Monocyte", GSE24759.meta$cellType, fixed = TRUE))
                   ]<-"Monocyte"
> GSE24759.meta$type[
   grep("Granulocyte (Neutrophil)", GSE24759.meta$cellType, fixed = TRUE)
                   ]<-"Granulocyte"
> GSE24759.meta$type[
   grep("Erythroid_CD34- CD71+", GSE24759.meta$cellType, fixed = TRUE)
                   ]<-"Nucleated Erythrocytes"
> GSE24759.meta.types<-levels(as.factor(GSE24759.meta$type))
> GSE24759.majorType.consensus<-sapply(GSE24759.meta.types, function(x){
     consensusFingerprint(GEO.fingerprint.matrix[,
         GSE24759.meta$GSM[GSE24759.meta$type %in% x]],
     threshold = threshold)
     })
> GSE24759.majorType.consensus[1:5, 1:2]
                                                 B-Cell
Glycolysis / Gluconeogenesis (KEGG)                  -1
Citrate cycle (TCA cycle) (KEGG)                     -1
Pentose phosphate pathway (KEGG)                     -1
Pentose and glucuronate interconversions (KEGG)      -1
Fructose and mannose metabolism (KEGG)               -1
                                                 CD4+_activated
Glycolysis / Gluconeogenesis (KEGG)                          -1
Citrate cycle (TCA cycle) (KEGG)                             -1
Pentose phosphate pathway (KEGG)                             -1
Pentose and glucuronate interconversions (KEGG)               0
Fructose and mannose metabolism (KEGG)                       -1


> GSE24759.majorType.dat <- phyDat(t(GSE24759.majorType.consensus), type = "USER", level
> GSE24759.majorType.dist <- dist.hamming(GSE24759.majorType.dat)
> # construct trees
> GSE24759.majorType.NJ.tree <- NJ(GSE24759.majorType.dist)
> GSE24759.majorType.parsimony.boot <- bootstrap.phyDat(
   GSE24759.majorType.dat, bs = 10, pratchet, start = GSE24759.majorType.NJ.tree, k = 50
```

```
   method = "sankoff", cost = CM, trace = 0, np = 1, all = TRUE)
> GSE24759.majorType.parsimony.boot<-c(
   GSE24759.majorType.parsimony.boot[
     lapply(GSE24759.majorType.parsimony.boot, class) == "phylo"
     ],
   unlist(GSE24759.majorType.parsimony.boot[
     lapply(GSE24759.majorType.parsimony.boot, class) == "multiPhylo"
     ],
          recursive = FALSE)
   )
> # Convert to cladewise ordering of edges
> GSE24759.majorType.parsimony.boot<-lapply(GSE24759.majorType.parsimony.boot, reorder,
> class(GSE24759.majorType.parsimony.boot)<-"multiPhylo"
> # Create consensus tree
> GSE24759.majorType.parsimony.consensus<-consensus(GSE24759.majorType.parsimony.boot, p
> # Add bootstrap scores
> GSE24759.majorType.parsimony.consensus$node.label<-round((100*prop.clades(
   GSE24759.majorType.parsimony.consensus, GSE24759.majorType.parsimony.boot)
                       )/length(GSE24759.majorType.parsimony.boot))
> for (i in 1:length(GSE24759.majorType.parsimony.boot)){
   GSE24759.majorType.parsimony.boot[[i]]$node.label<-(100*prop.clades(
     GSE24759.majorType.parsimony.boot[[i]], GSE24759.majorType.parsimony.boot)
                       )/length(GSE24759.majorType.parsimony.boot)
 }
```

As before, we can now either select the tree with highest summed bootstrap scores or the best parsimony scores. Here we will use the best parsimony score.

```
> GSE24759.majorType.bootstrap.scores<-data.frame(
   pScore = sapply(GSE24759.majorType.parsimony.boot, attr, "pscore"),
   sumBoot = sapply(GSE24759.majorType.parsimony.boot, function(x){
     sum(x$node.label)
     })
   )
> # Show top ordered trees
> head(GSE24759.majorType.bootstrap.scores[
   order(GSE24759.majorType.bootstrap.scores$pScore, -GSE24759.majorType.bootstrap.score
   ])
   pScore  sumBoot
4     560 484.6154
9     565 469.2308
8     565 461.5385
3     578 538.4615
11    580 530.7692
10    580 523.0769
```

```
> head(GSE24759.majorType.bootstrap.scores[
   order(GSE24759.majorType.bootstrap.scores$sumBoot, decreasing = TRUE),
   ])
   pScore  sumBoot
2     626 546.1538
6     630 546.1538
3     578 538.4615
13    581 538.4615
11    580 530.7692
10    580 523.0769
> # Select tree with best parsimony score
> GSE24759.majorType.parsimony.top<-GSE24759.majorType.parsimony.boot[[
   order(GSE24759.majorType.bootstrap.scores$pScore, -GSE24759.majorType.bootstrap.score
   ]]
> GSE24759.majorType.parsimony.top$node.label<-round(GSE24759.majorType.parsimony.top$no
> # Root trees to HSC
> try(GSE24759.majorType.parsimony.top<-root(
     GSE24759.majorType.parsimony.top, 7, resolve.root = TRUE))
> try(GSE24759.majorType.parsimony.consensus<-root(
     GSE24759.majorType.parsimony.consensus, 7, resolve.root = TRUE))


> plot(GSE24759.majorType.parsimony.top, show.node.label = FALSE, label.offset = 0)
> nodelabels(GSE24759.majorType.parsimony.top$node.label, bg = "white", cex = 0.75)


> plot(GSE24759.majorType.parsimony.consensus, show.node.label = FALSE, label.offset = 0
> nodelabels(GSE24759.majorType.parsimony.consensus$node.label, bg = "white", cex = 0.75
```

```
> GSE24759.majorType.inform<-rownames(GSE24759.majorType.consensus)[apply(GSE24759.major
> GSE7658.inform<-rownames(GSE7658.data)[apply(GSE7658.data, 1, sd) > 0]
> GSE6506.inform<-rownames(GSE6506.consensus)[apply(GSE6506.consensus, 1, sd) > 0]
> GSE24759.majorType.GSE7658.inform<-intersect(GSE24759.majorType.inform, GSE7658.inform
> GSE24759.majorType.GSE6506.inform<-intersect(GSE24759.majorType.inform, GSE6506.inform


> GSE7658.majorType.branches.inform<-optimP.branch(GSE7658.data[GSE24759.majorType.GSE76
                                        GSE24759.majorType.consensus[GSE24759.majorType.
                                        GSE24759.majorType.parsimony.top
                                        )


> l<-length(GSE7658.majorType.branches.inform)
> par(mfcol = c(((l %/% 3)+as.numeric((l %% 3 > 0))), ((l %/% 3)+as.numeric((l %% 3 > 0)
> for (i in 1:length(GSE7658.majorType.branches.inform)){
   plot(GSE7658.majorType.branches.inform[[i]],
        tip.color = GSE7658.majorType.branches.inform[[i]]$tip.color)
 }


> GSE6506.majorType.branches.inform<-optimP.branch(GSE6506.consensus[GSE24759.majorType.
                                        GSE24759.majorType.consensus[GSE24759.majorType.
                                        GSE24759.majorType.parsimony.top
                                        )


> l<-length(GSE6506.majorType.branches.inform)
> par(mfcol = c(((l %/% 3)+as.numeric((l %% 3 > 0))), ((l %/% 3)+as.numeric((l %% 3 > 0)
> for (i in 1:length(GSE6506.majorType.branches.inform)){
   plot(GSE6506.majorType.branches.inform[[i]],
        tip.color = GSE6506.majorType.branches.inform[[i]]$tip.color)
 }
```

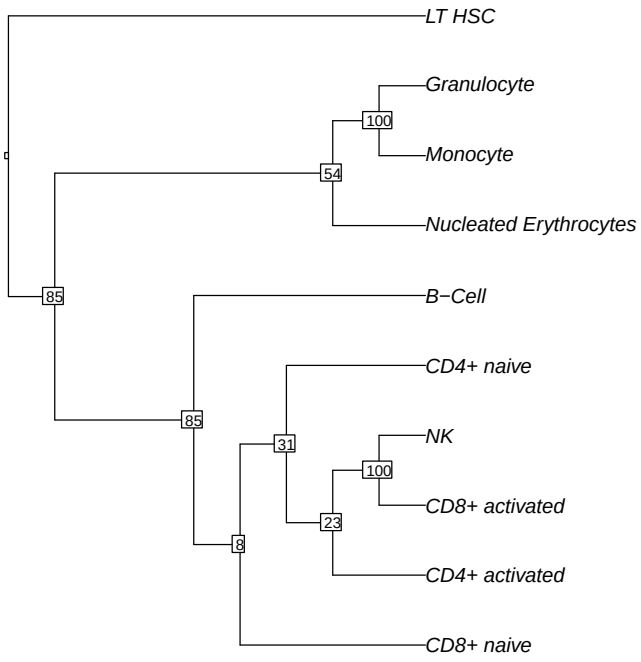

Figure 8: Maximum parsimony tree with highest bootstrap values for GSE24759 major cell types, numbers indicate bootstrap values

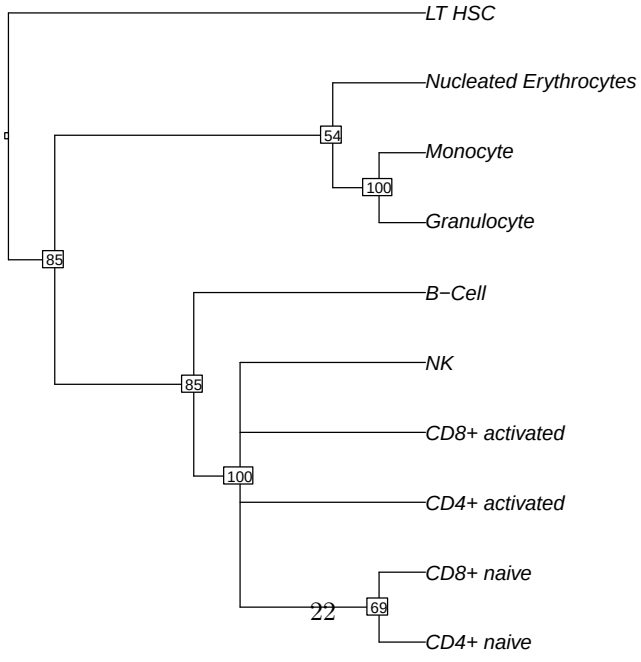

Figure 9: Consensus parsimony tree for GSE24759 major cell types from bootstraps, numbers indicate bootstrap values

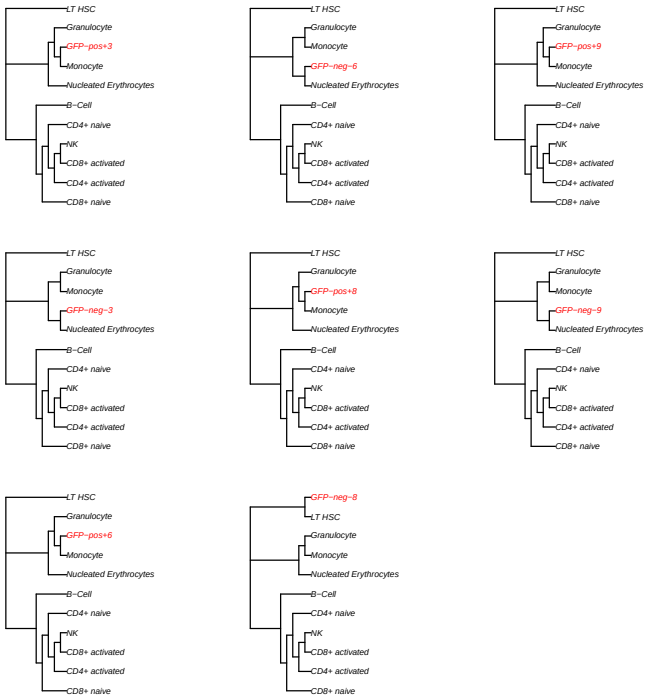

Figure 10: Maximum parsimony trees for each GSE7658 cell type integrated into the GSE24759 major types tree

Figure 11: Maximum parsimony trees for each GSE6506 cell type integrated into the GSE24759 major types tree